# DATABASE Laboratory Manual

Name: _____

Schedule: _____

CONTENTS

**Exercise Description**

**Read the General Information and the description of the lab tables before attempting to perform the exercises.**

Feel free to consult with the instructor if you need assistance or clarifications while you are formulating a query.

**Problem List** – Contains a listing of the problems and the expected result of the query for that exercise.

**Expected Result** – Shows the generated temporary table from the successful query. In most cases, the complete result is shown. In case where the result set is too large, an ellipses (...) is shown to indicate that there are additional rows in the result. The number of rows for the result set is shown so you may verify the result of your query.

**General Information**

This laboratory workbook provides the information necessary to complete the review lab exercises for the course.

This exercises will be done in-class environment. You may use any query

tools. Please take note that the emphasis of this course is on teaching **SQL**, not on tool use. Therefore, only the basics of the tool use is covered.

Write your solutions of your queries is a must.

**Table Relationships**

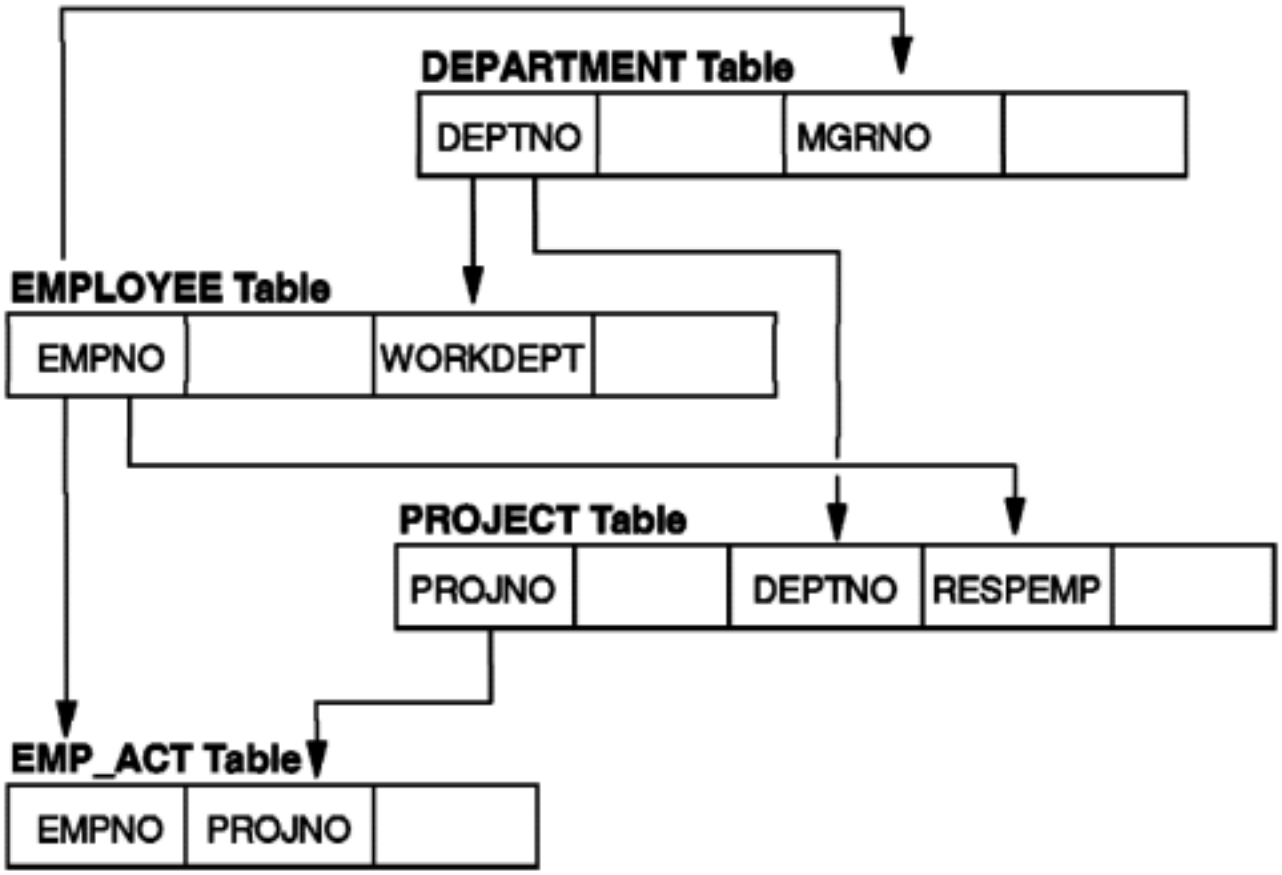


Figure 0-1. Table Relationships

**Note:**

This diagram illustrates the relationships between the tables used in the exercises for this course. The arrow lines show the connection of the tables using the primary key attribute to the foreign key attribute.

**Table Descriptions/Structures**

The tables are described in hierarchical order, as shown in the Table Relationships on the previous page.

**DEPARTMENT Table**

There is only one row/tuple in the DEPARTMENT table for each department in the company.

| Column Name | Meaning | Data Type | NULLS allowed |
|---|---|---|---|
| DEPTNO | Department Number | CHAR(3) | N |
| DEPTNAME | Department Name | VARCHAR(36) | N |
| MGRNO | Employee Number of the Responsible Manager | CHAR(6) | Y |
| ADMRDEPT | Department Number of the Department to which the Department reports | CHAR(3) | N |
| LOCATION | Location Number | CHAR(5) | Y |

**EMPLOYEE Table**

There is only one row/tuple in the EMPLOYEE table for each employees in the company.

| Column Name | Meaning | Data Type | NULLS allowed |
|---|---|---|---|
| EMPNO | Employee Number | CHAR(6) | N |
| FIRSTNME | First Name | VARCHAR(20) | N |
| MIDINIT | Middle Initial | CHAR(1) | N |
| LASTNAME | Last Name | VARCHAR(15) | N |
| WORKDEPT | Department in which the Employee Works | CHAR(3) | Y |
| PHONENO | Phone Number | CHAR(4) | Y |
| HIREDATE | Date of Hire | DATE | Y |
| JOB | Job | CHAR(8) | Y |
| EDLEVEL | Number of Years of Formal Education | SMALLINT | Y |
| SEX | Sex (M male, F female) | CHAR(1) | Y |
| BIRTHDATE | Date of Birth | DATE | Y |
| SALARY | Yearly Salary | DECIMAL(9, 2) | Y |
| BONUS | Yearly Bonus | DECIMAL(9, 2) | Y |
| COMM | Yearly Commission | DECIMAL(9, 2) | Y |

**PROJECT Table**

There is only one row/tuple in the PROJECT table for each project.

| Column Name | Meaning | Data Type | NULLS allowed |
|---|---|---|---|
| PROJNO | Project Number | CHAR(6) | N |
| PROJNAME | Project Name | VARCHAR(24) | N |
| DEPTNO | Responsible Department | CHAR(3) | N |
| RESPEMP | Employee Number of the Responsible Employee | CHAR(6) | N |
| PRSTAFF | Estimated Mean Staffing | DECIMAL(5, 2) | Y |
| PRSTDATE | Estimated Start Date | DATE | Y |
| PRENDATE | Estimated End Date | DATE | Y |
| MAJPROJ | Major Project for a Subproject | CHAR(6) | Y |

**EMP_ACT Table**

There is only one row/tuple in the PROJECT table for any employee or any project.

| Column Name | Meaning | Data Type | NULLS allowed |
|---|---|---|---|
| EMPNO | Employee Number of Employee Performing the Activity | CHAR(6) | N |
| PROJNO | Project Number | CHAR(6) | N |
| ACTNO | Activity Number | SMALLINT | N |
| EMPTIME | Proportion of Employee's Time Spent on Project | DECIMAL(5, 2) | Y |
| EMSTDATE | Date Activity Starts | DATE | Y |
| EMENDATE | Date Activity Ends | DATE | Y |

**Table Rows/Tuples**

**DEPARTMENT Table**

| DEPTNO | DEPTNAME | MGRNO | ADMRDEPT | LOCATION |
|---|---|---|---|---|
| A00 | SPIFFY COMPUTER SERVICE DIV. | 000010 | A00 | |
| B01 | PLANNING | 000020 | A00 | |
| C01 | INFORMATION CENTER | 000030 | A00 | |
| D01 | DEVELOPMENT CENTER | - | A00 | |
| D11 | MANUFACTURING SYSTEMS | 000060 | D01 | |
| D21 | ADMINISTRATION SYSTEMS | 000070 | D01 | |
| E01 | SUPPORT SERVICES | 000050 | A00 | |
| E11 | OPERATIONS | 000090 | E01 | |
| E21 | SOFTWARE SUPPORT | 000100 | E01 | |

**EMPLOYEE Table**

| EMPNO | FIRSTNME | MIDINIT | LASTNAME | WORKDEPT | PHONENO | HIREDATE |
|---|---|---|---|---|---|---|
| 000010 | CHRISTINE | I | HAAS | A00 | 3978 | 1965-01-01 |
| 000020 | MICHAEL | L | THOMPSON | B01 | 3476 | 1973-10-10 |
| 000030 | SALLY | A | KWAN | C01 | 4738 | 1975-04-05 |
| 000050 | JOHN | B | GEYER | E01 | 6789 | 1949-08-17 |
| 000060 | IRVING | F | STERN | D11 | 6423 | 1973-09-14 |
| 000070 | EVA | D | PULASKI | D21 | 7831 | 1980-09-30 |
| 000090 | EILEEN | W | HENDERSON | E11 | 5498 | 1970-08-15 |
| 000100 | THEODORE | Q | SPENSER | E21 | 0972 | 1980-06-19 |
| 000110 | VINCENZO | G | LUCCHESI | A00 | 3490 | 1958-05-16 |
| 000120 | SEAN | | O'CONNELL | A00 | 2167 | 1963-12-05 |
| 000130 | DOLORES | M | QUINTANA | C01 | 4578 | 1971-07-28 |
| 000140 | HEATHER | A | NICHOLLS | C01 | 1793 | 1976-12-15 |
| 000150 | BRUCE | | ADAMSON | D11 | 4510 | 1972-02-12 |
| 000160 | ELIZABETH | R | PIANKA | D11 | 3782 | 1977-10-11 |
| 000170 | MASATOSHI | J | YOSHIMURA | D11 | 2890 | 1978-09-15 |
| 000180 | MARILYN | S | SCOUTTEN | D11 | 1682 | 1973-07-07 |
| 000190 | JAMES | H | WALKER | D11 | 2986 | 1974-07-26 |
| 000200 | DAVID | | BROWN | D11 | 4501 | 1966-03-03 |
| 000210 | WILLIAM | T | JONES | D11 | 0942 | 1979-04-11 |
| 000220 | JENNIFER | K | LUTZ | D11 | 0672 | 1968-08-29 |
| 000230 | JAMES | J | JEFFERSON | D21 | 4265 | 1966-11-21 |
| 000240 | SALVATORE | M | MARINO | D21 | 3780 | 1979-12-05 |
| 000250 | DANIEL | S | SMITH | D21 | 0961 | 1969-10-30 |
| 000260 | SYBIL | V | JOHNSON | D21 | 8953 | 1975-09-11 |
| 000270 | MARIA | L | PEREZ | D21 | 9001 | 1980-09-30 |
| 000280 | ETHEL | R | SCHNEIDER | E11 | 8997 | 1967-03-24 |
| 000290 | JOHN | R | PARKER | E11 | 4502 | 1980-05-30 |
| 000300 | PHILIP | X | SMITH | E11 | 2095 | 1972-06-19 |
| 000310 | MAUDE | F | SETRIGHT | E11 | 3332 | 1964-09-12 |
| 000320 | RAMLAL | V | MEHTA | E21 | 9990 | 1965-07-07 |
| 000330 | WING | | LEE | E21 | 2103 | 1976-02-23 |
| 000340 | JASON | R | GOUNOT | E21 | 5698 | 1947-05-05 |

**EMPLOYEE Table (continued)**

| JOB | EDLEVEL | SEX | BIRTHDATE | SALARY | BONUS | COMM |
|---|---|---|---|---|---|---|
| PRES | 18 | F | 1933-08-14 | 52750.00 | 1000.00 | 4220.00 |
| MANAGER | 18 | M | 1948-02-02 | 41250.00 | 800.00 | 3300.00 |
| MANAGER | 20 | F | 1941-05-11 | 38250.00 | 800.00 | 3060.00 |
| MANAGER | 16 | M | 1925-09-15 | 40175.00 | 800.00 | 3214.00 |
| MANAGER | 16 | M | 1945-07-07 | 32250.00 | 600.00 | 2580.00 |
| MANAGER | 16 | F | 1953-05-26 | 36170.00 | 700.00 | 2893.00 |
| MANAGER | 16 | F | 1941-05-15 | 29750.00 | 600.00 | 2380.00 |
| MANAGER | 14 | M | 1956-12-18 | 26150.00 | 500.00 | 2092.00 |
| SALESREP | 19 | M | 1929-11-05 | 46500.00 | 900.00 | 3720.00 |
| CLERK | 14 | M | 1942-10-18 | 29250.00 | 600.00 | 2340.00 |
| ANALYST | 16 | F | 1925-09-15 | 23800.00 | 500.00 | 1904.00 |
| ANALYST | 18 | F | 1946-01-19 | 28420.00 | 600.00 | 2274.00 |
| DESIGNER | 16 | M | 1947-05-17 | 25280.00 | 500.00 | 2022.00 |
| DESIGNER | 17 | F | 1955-04-12 | 22250.00 | 400.00 | 1780.00 |
| DESIGNER | 16 | M | 1951-01-05 | 24680.00 | 500.00 | 1974.00 |
| DESIGNER | 17 | F | 1949-02-21 | 21340.00 | 500.00 | 1707.00 |
| DESIGNER | 16 | M | 1952-06-25 | 20450.00 | 400.00 | 1636.00 |
| DESIGNER | 16 | M | 1941-05-29 | 27740.00 | 600.00 | 2217.00 |
| DESIGNER | 17 | M | 1953-02-23 | 18270.00 | 400.00 | 1462.00 |
| DESIGNER | 18 | F | 1948-03-19 | 29840.00 | 600.00 | 2387.00 |
| CLERK | 14 | M | 1935-05-30 | 22180.00 | 400.00 | 1774.00 |
| CLERK | 17 | M | 1954-03-31 | 28760.00 | 600.00 | 2301.00 |
| CLERK | 15 | M | 1939-11-12 | 19180.00 | 400.00 | 1534.00 |
| CLERK | 16 | F | 1936-10-05 | 17250.00 | 300.00 | 1380.00 |
| CLERK | 15 | F | 1953-05-26 | 27380.00 | 500.00 | 2190.00 |
| OPERATOR | 17 | F | 1936-03-28 | 26250.00 | 500.00 | 2100.00 |
| OPERATOR | 12 | M | 1946-07-09 | 15340.00 | 300.00 | 1227.00 |
| OPERATOR | 14 | M | 1936-10-27 | 17750.00 | 400.00 | 1420.00 |
| OPERATOR | 12 | F | 1931-04-21 | 15900.00 | 300.00 | 1272.00 |
| FIELDREP | 16 | M | 1932-08-11 | 19950.00 | 400.00 | 1596.00 |
| FIELDREP | 14 | M | 1941-07-18 | 25370.00 | 500.00 | 2030.00 |
| FIELDREP | 16 | M | 1926-05-17 | 23840.00 | 500.00 | 1907.00 |

**PROJECT Table**

| PROJNO | PROJNAME | DEPT NO | RESPEMP | PR STAFF | PRSTDATE | PRENDATE | MAJPROJ |
|---|---|---|---|---|---|---|---|
| AD3100 | ADMIN SERVICES | D01 | 000010 | 6.50 | 1982-01-01 | 1983-02-01 | - |
| AD3110 | GENERAL AD SYSTEMS | D21 | 000070 | 6.00 | 1982-01-01 | 1983-02-01 | AD3100 |
| AD3111 | PAYROLL PROGRAMMING | D21 | 000230 | 2.00 | 1982-01-01 | 1983-02-01 | AD3110 |
| AD3112 | PERSONNEL PROGRAMMG | D21 | 000250 | 1.00 | 1982-01-01 | 1983-02-01 | AD3110 |
| AD3113 | ACCOUNT.PROGRAMMING | D21 | 000270 | 2.00 | 1982-01-01 | 1983-02-01 | AD3110 |
| IF1000 | QUERY SERVICES | C01 | 000030 | 2.00 | 1982-01-01 | 1983-02-01 | - |
| IF2000 | USER EDUCATION | C01 | 000030 | 1.00 | 1982-01-01 | 1983-02-01 | - |
| MA2100 | WELD LINE AUTOMATION | D01 | 000010 | 12.00 | 1982-01-01 | 1983-02-01 | - |
| MA2110 | W L PROGRAMMING | D11 | 000060 | 9.00 | 1982-01-01 | 1983-02-01 | MA2100 |
| MA2111 | W L PROGRAM DESIGN | D11 | 000220 | 2.00 | 1982-01-01 | 1982-12-01 | MA2110 |
| MA2112 | W L ROBOT DESIGN | D11 | 000150 | 3.00 | 1982-01-01 | 1982-12-01 | MA2110 |
| MA2113 | W L PROD CONT PROGS | D11 | 000160 | 3.00 | 1982-02-15 | 1982-12-01 | MA2110 |
| OP1000 | OPERATION SUPPORT | E01 | 000050 | 6.00 | 1982-01-01 | 1983-02-01 | - |
| OP1010 | OPERATION | E11 | 000090 | 5.00 | 1982-01-01 | 1983-02-01 | OP1000 |
| OP2000 | GEN SYSTEMS SERVICES | E01 | 000050 | 5.00 | 1982-01-01 | 1983-02-01 | - |
| OP2010 | SYSTEMS SUPPORT | E21 | 000100 | 4.00 | 1982-01-01 | 1983-02-01 | OP2000 |
| OP2011 | SCP SYSTEMS SUPPORT | E21 | 000320 | 1.00 | 1982-01-01 | 1983-02-01 | OP2010 |
| OP2012 | APPLICATIONS SUPPORT | E21 | 000330 | 1.00 | 1982-01-01 | 1983-02-01 | OP2010 |
| OP2013 | DB/DC SUPPORT | E21 | 000340 | 1.00 | 1982-01-01 | 1983-02-01 | OP2010 |
| PL2100 | WELD LINE PLANNING | B01 | 000020 | 1.00 | 1982-01-01 | 1982-09-15 | MA2100 |

**EMP_ACT Table**

| EMPNO | PROJNO | ACTNO | EMPTIME | EMSTDATE | EMENDATE |
|---|---|---|---|---|---|
| 000010 | AD3100 | 10 | 0.50 | 1982-01-01 | 1982-07-01 |
| 000070 | AD3110 | 10 | 1.00 | 1982-01-01 | 1983-02-01 |
| 000230 | AD3111 | 60 | 1.00 | 1982-01-01 | 1982-03-15 |
| 000230 | AD3111 | 60 | 0.50 | 1982-03-15 | 1982-04-15 |
| 000230 | AD3111 | 70 | 0.50 | 1982-03-15 | 1982-10-15 |
| 000230 | AD3111 | 80 | 0.50 | 1982-04-15 | 1982-10-15 |
| 000230 | AD3111 | 180 | 1.00 | 1982-10-15 | 1983-01-01 |
| 000240 | AD3111 | 70 | 1.00 | 1982-02-15 | 1982-09-15 |
| 000240 | AD3111 | 80 | 1.00 | 1982-09-15 | 1983-01-01 |
| 000250 | AD3112 | 60 | 0.50 | 1982-02-01 | 1982-03-15 |
| 000250 | AD3112 | 60 | 1.00 | 1982-01-01 | 1982-02-01 |
| 000250 | AD3112 | 60 | 1.00 | 1983-01-01 | 1983-02-01 |
| 000250 | AD3112 | 60 | 0.50 | 1982-12-01 | 1983-01-01 |
| 000250 | AD3112 | 70 | 1.00 | 1982-03-15 | 1982-08-15 |
| 000250 | AD3112 | 70 | 0.50 | 1982-02-01 | 1982-03-15 |
| 000250 | AD3112 | 70 | 0.25 | 1982-08-15 | 1982-10-15 |
| 000250 | AD3112 | 80 | 0.25 | 1982-08-15 | 1982-10-15 |
| 000250 | AD3112 | 80 | 0.50 | 1982-10-15 | 1982-12-01 |
| 000250 | AD3112 | 180 | 0.50 | 1982-08-15 | 1983-01-01 |
| 000260 | AD3113 | 70 | 0.50 | 1982-06-15 | 1982-07-01 |
| 000260 | AD3113 | 70 | 1.00 | 1982-07-01 | 1983-02-01 |
| 000260 | AD3113 | 80 | 1.00 | 1982-01-01 | 1982-03-01 |
| 000260 | AD3113 | 80 | 0.50 | 1982-03-01 | 1982-04-15 |
| 000260 | AD3113 | 180 | 0.50 | 1982-03-01 | 1982-04-15 |
| 000260 | AD3113 | 180 | 0.50 | 1982-06-01 | 1982-07-01 |
| 000260 | AD3113 | 180 | 1.00 | 1982-04-15 | 1982-06-01 |
| 000270 | AD3113 | 60 | 0.25 | 1982-09-01 | 1982-10-15 |
| 000270 | AD3113 | 60 | 1.00 | 1982-04-01 | 1982-09-01 |
| 000270 | AD3113 | 60 | 0.50 | 1982-03-01 | 1982-04-01 |
| 000270 | AD3113 | 70 | 0.75 | 1982-09-01 | 1982-10-15 |
| 000270 | AD3113 | 70 | 1.00 | 1982-10-15 | 1983-02-01 |
| 000270 | AD3113 | 80 | 1.00 | 1982-01-01 | 1982-03-01 |
| 000270 | AD3113 | 80 | 0.50 | 1982-03-01 | 1982-04-01 |
| 000030 | IF1000 | 10 | 0.50 | 1982-06-01 | 1983-01-01 |
| 000130 | IF1000 | 90 | 1.00 | 1982-01-01 | 1982-10-01 |
| 000130 | IF1000 | 100 | 0.50 | 1982-10-01 | 1983-01-01 |
| 000140 | IF1000 | 90 | 0.50 | 1982-10-01 | 1983-01-01 |

**Exercise 1. SQL DDL – Data Definition Language**

**What is this Exercise is About**

This exercise provides a knowledge to code SQL statements in order to perform Database operations using data definition language.

## What You Should Be Able To Do

At the end of the lab exercises, you should be able to:

- Code CREATE statement to create Database and Database objects such as tables, views, schemas, etc.
- Code ALTER statement to modify table structure.
- Code DROP statement to remove Database from the server machine.
- Code RENAME statement to change table names.

## Introduction

See the data model at the start of this exercise guide to get the table names, column names and descriptions for each table.

## Supporting Resources

- Instructor's instructional materials
- SQL References (Online resources, handouts, notes, etc.)

***Problem 1:***

Create the Database that is specified by your instructor in the default drive.

***Problem 2:***

Verify if the Database created in problem 1 was successfully created, write the details of the result below.

***Problem 3:***

Create the table DEPARTMENT.

***Problem 4:***

Create the table EMPLOYEE.

***Problem 5:***

Create the table PROJECT.

***Problem 6:***

Create the table EMP_ACT.

***Problem 7:***

Verify if the tables were successfully created.

***Problem 8:***

Give the result of the above SQL statement.

***Problem 9:***

Give the SQL statement to view the structure of DEPARTMENT table.

***Problem 10:***

Give the result of the above SQL statement.

***Problem 11:***

Modify the structure of the DEPARTMENT table to define MGRNO as the foreign key used to relate to the EMPLOYEE table.

***Problem 12:***

Modify the structure of the DEPARTMENT table to change the data type of the attribute/column name LOCATION to VARCHAR 30.

***Problem 13:***

Change the name of EMP_ACT table to ACCOUNT.

Is the SQL statement successful?

Why?

***Problem 14:***

Perform the given SQL command.

```
CREATE TABLE mytbl (
    MyNumber SMALLINT NOT NULL,
    MyName VARCHAR (30) NOT NULL);
```

Change the name of table mytbl to newtbl

Is the SQL statement successful?

## Exercise 2. SQL DML – Data Manipulation Language

### What is this Exercise is About

This exercise provides a knowledge to code SQL statements in order to perform Database operations using data manipulation language.

### What You Should Be Able To Do

At the end of the lab exercises, you should be able to:

- ☐ Code INSERT statement to store rows or records into tables of the Database.
- ☐ Code UPDATE statement to modify rows or records of the table.
- ☐ Code DELETE statement to remove rows or records from the tables of the Database.

### Introduction

See the data model at the start of this exercise guide to get the table names, column names and descriptions for each table.

### Supporting Resources

- ☐ Instructor's instructional materials
- ☐ SQL References (Online resources, hand outs, notes, etc.)

***Problem 1:***

Store the first 5 rows/tuples of DEPARTMENT table.

***Problem 2:***

Store the first 5 rows/tuples of EMPLOYEE table.

***Problem 3:***

Store the first 5 rows/tuples of PROJECT table.

***Problem 4:***

Store the first 5 rows/tuples of EMP_ACT table.

***Problem 5:***

Update the rows/tuples of the DEPARTMENT table to store the values of the attribute LOCATION.

**DEPTNO LOCATION**

- - - - - - - - - - - - - - - - - -

A00 R. Palma St. Cebu City

B01 CCICT Bldg. 2nd Flr.

C01 CCICT Bldg. 2nd Flr.

D01 CCICT Bldg. 3rd Flr.

D11 CCICT Bldg. 1st Flr.

***Problem 6:***

Jason R. Gounot with an employee number 000340 retired from the company, perform the necessary table adjustment.

**Exercise 3. SQL DQL – SIMPLE QUERY**

**What is this Exercise is About**

This exercise provides a knowledge to code SQL statements in order to perform Database operations using data query language.

**What You Should Be Able To Do**

At the end of the lab exercises, you should be able to:

- Code SELECT statements using four clauses of an SQL SELECT

statement.

- Use SELECT statement to:
  - ~ Retrieve all rows/records of a table.
  - ~ Retrieve specific columns/attributes of a table.
  - ~ Retrieve specific rows/records based on some relational or logical expressions.
- Code SELECT statement using keywords BETWEEN, IN, LIKE and DISTINCT.
- Produce a result table in which rows/records are sorted in a desired sequence.

**Introduction**

See the data model at the start of this exercise guide to get the table names, column names and descriptions for each table.

**Supporting Resources**

- Instructor's instructional materials
- SQL References (Online resources, hand outs, notes, etc.)

**Problem List With Expected Results**

***Problem 1:***

List employee number, last name, data of birth, and salary for all employees who make more than $30,000.00 a year. Sequence the results in descending order by salary.

***Result list:***

***Problem 2:***

List last name, first name, and the department number for all employees. The listing should be ordered by descending department numbers. Within the same department, the last name should be sorted in descending order.

***Result list:***

***Problem 3:***

List the different education levels in the company in descending order. List only one occurrence of duplicate result rows.

***Result list:***

***Problem 4:***

List employees, by employee number, and their assigned projects, by project number. Display only those employees with an employee number less than or equal to 100. List only one occurrence of duplicate result rows. Sort the result rows by employee number.

(Use the EMP_ACT table)

***Result list:***

***Problem 5:***

List last name, salary, and bonus of all male employees.

***Result list:***

***Problem 6:***

List last name, salary, and commission for all employees with a salary greater than $29,000.00 and was hired after 1979.

***Result list:***

***Problem 7:***

List last name, salary, bonus, and commission for all employees with a salary greater than $22,000.00 and a bonus of $400.00, or for all employees with a bonus of $500.00 and a commission lower than

$1,900.00. The list should be ordered by last name.

***Result list:***

***Problem 8:***

List last name, salary, bonus, and commission for all employees with a salary greater than $22,000.00, a bonus of $400.00, or $500.00 and a commission less than $1,900.00. The list should be ordered by last name.

***Result list:***

***Problem 9:***

List all departments that have 1 as the middle character in their department number.

***Result list:***

***Problem 10:***

Using the EMP_ACT table, for all projects that have a project number beginning with AD and have activities 10, 80, and 180 associated with them, list the following:

- Project number
- Activity number
- Starting date of activity
- Ending date of activity

Order the list by activity number within project number.

***Result list:***

***Problem 11:***

List manager number and the department number for all departments to which a manager has been assigned.

The list should be ordered by manager number.

***Result list:***

***Problem 12:***

List employee number, last name, salary, and bonus for all employees that have a bonus ranging from $800.00 to $1,000.00.

Sort the report by employee number within bonus, lowest bonus first.

***Result list:***

***Problem 13:***

List employee number, last name, salary, and department number for all employees in departments A00 through C01 (inclusive).

Order the results alphabetically by last name and employee number.

***Result list:***

***Problem 14:***

List all projects that have SUPPORT as part of the project name. Order

the result list by project number.

***Result list:***

***Problem 15:***

List department number, department name, and manager number for all departments that have 1 as the middle character in the department number. Order the cursor list by department number.

**Exercise 4. SQL DQL – JOIN QUERY**

**What is this Exercise is About**

This exercise provides a knowledge to code SQL statements to retrieve data from multiple tables using JOIN query.

**What You Should Be Able To Do**

At the end of the lab exercises, you should be able to:

- Code JOIN statements applying four types of JOIN clause to retrieve data from multiple tables.
- Apply the four types of JOIN clauses:
  - ~ Inner Join (Default Join Query).
  - ~ Cross Join or Cartesian Product (Must be avoided).
  - ~ Self-Join (Joining Table to Itself).
  - ~ Outer Join.

    Left Outer Join Right

    Outer Join Full Outer

    Join

**Introduction**

See the data model at the start of this exercise guide to get the table names, column names and descriptions for each table.

**Supporting Resources**

- [ ] Instructor's instructional materials
- [ ] SQL References (Online resources, hand outs, notes, etc.)

***Problem 1:***

Produce a report that lists employee's last name, first name, and department names. Sequence the result list on first name within last name, within department name.

***Result list:***

***Problem 2:***

Modify the previous report to include job. Also, list data for only departments between A02 and D22, and exclude the managers from the list. Sequence the

report on first name within last name, within job, within department name.

***Result list:***

***Problem 3:***

List the name of each department and the last name and first name of its manager. Sequence the list by department name. Use EMPNO and MGRNO columns to relate the two tables. Sequence the result rows by department name.

***Result list:***

***Problem 4:***

Modify the previous report (Query #3) by using WORKDEPT and DEPTNO as the joining predicate. Include a local predicate that looks for people whose job is manager.

Are the results from both queries the same? .

Why?

***Result list:***

***Problem 5:***

For all projects that have a project number beginning with AD, list project number, project name, and activity number. List identical rows only once. Order the list by project number and then by activity number.

***Result list:***

***Problem 6:***

Which employees are assigned to project number AD3113? List employee number, last name, and project number. Order the list by employee number and then by project number. List only one occurrence of duplicate result rows.

***Result list:***

***Problem 7:***

Which activities began on October 1, 1982? For each of these activities, list the employee number of the person performing the activity, the project number, the project name, activity number, and the starting date of the activity. Order the list by project number, then by employee number, and then by activity number.

***Result list:***

***Problem 8:***

Display department number, last name, project name, and activity number for activities performed by employees in department A00. Sequence the result first by project name and then by activity number.

***Result list:***

***Problem 9:***

List department number, last name, project name, and activity number for those employees in work departments A00 through C01. Suppress identical rows. Sort the list by department number, last name, and activity number.

***Result list:***

***Problem 10:***

The second line manager needs a list of activities which began on October 15, 1982 or thereafter. For these activities, list the activity number, the manger number of the manager assigned to the project, the starting date of the activity, the project number, and the last name of the employee performing the activity. The list should be ordered by activity number and then by the activity start date.

***Result list:***

***Problem 11:***

Which employees in department A00 were hired before their manager? List department number, the manager's last name, the employee's last name, and the hiring dates of both the manager and the employee.

Order the list by the employee's last name.

***Result list:***

## Exercise 5. SQL DQL – SCALAR FUNCTION AND ARITHMETIC QUERY

### What is this Exercise is About

This exercise provides a knowledge to code SQL statements using SCALAR functions and ARITHMETIC expressions.

### What You Should Be Able To Do

At the end of the lab exercises, you should be able to:

- Code queries that uses scalar functions.
- Code queries by using calculated expressions in the SELECT list and in the WHERE clause.
- Use basic and advance SCALAR function queries.
- Use the CONCAT operator in queries.

### Introduction

See the data model at the start of this exercise guide to get the table names, column names and descriptions for each table.

### Supporting Resources

- Instructor's instructional materials

SQL References (Online resources, handouts, notes, etc.)

***Problem 1:***

For employees, whose salary is increased by 5 percent, is less than or equal to $20,000.00, list the following:

- Last name
- Current Salary
- Salary increased by 5 percent
- Monthly salary increased by 5 percent

Use the following names for the two generated columns:

INC-Y-SALARY and INC-M-SALARY, Use the proper conversion function to display the increased salary and monthly salary with two digits to the right of the decimal point. Sort the results by annual salary.

***Result list:***

***Problem 2:***

All employees with an education level of 18 or 20 will receive a salary increase of $1,200.00 and their bonus will be cut in half. List last name, education level, new salary, and new bonus for these employees.
Display the new bonus with two digits to the right of the decimal point. Use the column names NEW-SALARY and NEW-BONUS for the generated columns.

Employees with an education level of 20 should be listed first. For employees with the same education level, sort the list by salary.

***Result list: Problem 3:***

The salary will be decreased by $1,000.00 for all employees matching the following criteria:

***Result list:***

• They belong to department D11
• Their salary is more than or equal to 80 percent of $20,000.00 • Their salary is less than or equal to 120 percent of $20,000.00

Use the name DECR-SALARY for the generated column. List department number, last name, salary, and decreased salary. Sort the result list by salary.

***Problem 4:***

Produce a list of all employees in department D11 that have an income (sum of salary, bonus, and commission) that is greater than their salary increased by 10 percent.
Name the generated column INCOME. List department number, last name, and income. Sort the result set in descending order by income. For this problem, assume that all employees have non-null salaries, bonus, and commission.

***Result list:***

***Problem 5:***

List all departments that have no managers assigned. List department number, department name, and manager number. Replace unknown manager numbers with the word UKNOWN and name the column MGRNO.

***Result list:***

***Problem 6:***

List the project number and major project number for all projects that have a project number beginning with MA. If the major project number is unknown, display the text **'MAIN PROJECT'**. Name the derived column MAJOR PROJECT. Sequence the results by PROJNO.

***Result list:***

***Problem 7:***

List all employees who were younger than 25 years old when they joined the company.
List their employee number, last name, and age when they joined the company, Name the derived column AGE. Sort the result by age and then by employee number.

***Result list:***

***Problem 8:***

Provide a list of all projects which ended on December 1, 1982. Display the year and the month of the starting date and the project number. Sort the result by project number.
Name the derived column YEAR and MONTH.

***Result list:***

***Problem 9:***

List the project number and duration, in weeks, for all projects that have a project number beginning with MA. The duration should be rounded and displayed with one decimal position.

Name the derived column WEEKS. Order the list by project number.

***Result list:***

***Problem 10:***

For project that have a project number beginning with MA, list the project number, project ending date, and a modified ending date, assuming the projects will be delayed by 10 percent.
Rename the column PRENDATE to ESTIMATED and the derived column EXPECTED. Order the list by project number.

***Result list:***

***Problem 11:***

How many days are there between the first manned landing on the moon (July 20, 1969) and the first day of the year 2000? Since no columns from a specific table are used in this problem, you can use any table in your Database. Include a WHERE clause that derives a single result row.

Name the derived column DAYS.

***Result list***

## Exercise 6. SQL DQL – COLUMN FUNCTION AND GROUP BY WITHOUT OR WITH HAVING CLAUSE

### What is this Exercise is About

This exercise provides a knowledge to code SQL statements using COLUMN functions and GROUP BY with or without HAVING clause.

### What You Should Be Able To Do

At the end of the lab exercises, you should be able to:

- Code queries that uses column functions.
- Code queries with complete SQL clauses.
- Code queries that use GROUP BY clause to group values together as one row.
- Code queries that include HAVING clause to produce a cursor with grouped rows.

### Introduction

See the data model at the start of this exercise guide to get the table names, column names and descriptions for each table.

### Supporting Resources

- Instructor's instructional materials
- SQL References (Online resources, handouts, notes, etc.)

***Problem 1:***

For all departments, display department number and the sum of all salaries for each department. Name the derived column SUM_SALARY.

***Result list:***

***Problem 2:***

For all departments, display department number and the number of employees in each department.

Name the derived column EMP_COUNT.

***Result list:***

***Problem 3:***

Display those departments which have more than 3 employees.

***Result list:***

***Problem 4:***

For all departments with at least one designer, display the number of designers and

the department number. Name the derived column DESIGNER.

***Result list:***

***Problem 5:***

Show the average salary for men and the average salary for women in each department. Display the work department, sex, the average salary, the average bonus, the average commission, and the number of people in each group. Include only those groups that have two or more people. Show only two decimal places in the averages.

Use the following names for the derived columns: AVG-SALARY, AVG- BONUS, AVG-COMM, and COUNT

***Result list:***

Display the average bonus and average commission for all departments with an average bonus greater than $500.00 and an average commission greater than $2,000.00. Display all averages with two digits to the right of the decimal
point. Use the column headings AVG-BONUS and AVG-COMM for the derived columns.

***Problem 6: Result list:***

**Exercise 7. SQL DQL – UNION QUERY**

## What is this Exercise is About

This exercise provides a knowledge to code SQL statements using UNION statement.

## What You Should Be Able To Do

At the end of the lab exercises, you should be able to:

- Code queries that uses UNION statement to combine multiple cursors or result table from multiple SELECT statements.

## Introduction

See the data model at the start of this exercise guide to get the table names, column names and descriptions for each table.

## Supporting Resources

- Instructor's instructional materials
- SQL References (Online resources, hand outs, notes, etc.)

***Problem 1:***

List the names and salaries for the non-managers working in department D21 showing the effects of a 10 percent raise. Use the following output as guide. Apply an appropriate ORDER BY clause to achieve the required results. Use the column headings as shown in the result list below.

***Result list:***

***Problem 2:***

List the department number, employee number, and salaries for all employees in

department A00.

For the last line of the report, display the sum of all the salaries

***Result list:***

***Problem 3:***

For departments A00, B01, and C01, list the projects assigned to them and the employees in each department. The output should consist of up to three types of lines for each department as follows:
See result list for clarifications of the problem. First

line (one per department)

- Department number
- Text: DEPARTMENT
- Department name

Second line(s) (if data are available – one line per project)

- Department number
- Project number
- Project name

Subsequent line(s) (if data are available – one line per employee)

- Department number
- Employee number
- Last name

***Result list:***

***Problem 4:***

For all projects that have a project number that begins with IF, display the following:

First line:

- Text: PROJECT
- Project number
- The employee number of the employee responsible for the project • Estimated starting date
- Estimated ending date

Subsequent line(s) (one per employee working on the project)

- Project number
- The employee number of the employee performing the activity
- Activity starting date
- Activity ending date

Sequence the results by the project number, then by the employee number, and finally by the starting date.

***Result list:***

## Exercise 8. SQL DQL – SUBQUERY

### What is this Exercise is About

This exercise provides a knowledge to code SQL statements using Subqueries.

### What You Should Be Able to Do

At the end of the lab exercises, you should be able to:

- ☐ Code basic subqueries - A query that is place inside another query.
- ☐ Code complex subqueries by using the keyword **EXISTS** and **IN**.

### Introduction

See the data model at the start of this exercise guide to get the table names, column names and descriptions for each table.

### Supporting Resources

- ☐ Instructor's instructional materials
- ☐ SQL References (Online resources, hand outs, notes, etc.)

***Problem 1:***

List those employees that have a salary which is greater than or equal to the average salary of all employees plus $500.00.
Display department number, employee number, last name, and salary. Sort the list by the department number and employee number.

***Result list:***

***Problem 2:***

List employee number and last name of all employees not assigned to any projects. This means that the table EMP_ACT does not contain a row with their employee number.

***Result list:***

***Problem 3:***

List project number and duration (in days) of the project with the shortest duration. Name the derived column DAYS.

***Result list:***

***Problem 4:***

List department number, department name, last name, and first name of those employees in departments that have only male employees.

***Result list:***

***Problem 5:***

We want to do a salary analysis for people that have the same job and education level as the employee Stern. Show the last name, job, edlevel, the number of years they've worked as of January 1, 2000, and their salary. Name the derived column YEARS. Sort the listing by highest salary first.

***Result list:***

## Exercise 9. SQL DQL – QUERY WITH INTERSECT AND EXCEPT STATEMENTS

### What is this Exercise is About

This exercise provides a knowledge to code SQL statements using INTERSECT and EXCEPT keywords.

### What You Should Be Able To Do

t the end of the lab exercises, you should be able to:

- Code queries with INTERSECT and EXCEPT keywords.
- Contrast between INTERSECT and EXCEPT keywords.

### Introduction

See the data model at the start of this exercise guide to get the table names, column names and descriptions for each table.

### Supporting Resources

- Instructor's instructional materials
- SQL References (Online resources, handouts, notes, etc.)

***Problem 1:***

Open 3 separate command editors.

In the first command editor, code the query that list the last name, education level, and job for all ANALYST employees.

***Result list:***

| LASTNAME | EDLEVEL | JOB |
|---|---|---|
| QUINTANA | 16 | ANALYST |
| NICHOLLS | 18 | ANALYST |
| NATZ | 18 | ANALYST |

In the second command editor, code the query that list last name, education level, and job for employees, whose education level is 18.

***Result list:***

| LASTNAME | EDLEVEL | JOB |
|---|---|---|
| HAAS | 18 | PRES |
| THOMPSON | 18 | MANAGER |
| NICHOLLS | 18 | ANALYST |
| LUTZ | 18 | DESIGNER |
| HEMMINGER | 18 | SALESREP |
| NATZ | 18 | ANALYST |
| JOHN | 18 | DESIGNER |

In the third command editor, code the query that combines your first query from the command editor 1 and your second query from the command editor 2 using INTERSECT keyword.

**Explain the cursor (result table)**

**ANSWER:**

- It displays only the rows that are exactly the same in both result sets — meaning the employee must be an ANALYST and must have an education level of 18. From the data, only NICHOLLS and NATZ meet both conditions. That's why the result table only includes those two rows. This demonstrates how INTERSECT is used to find common records between two SQL queries.

***Problem 2:***

Open 3 separate command editors.

In the first command editor, code the query that list employee number, last name, job, work department, project number, and project name for all employees who works at departments A00 through D11.

***Result list:***

| EMPNO | LASTNAME | JOB | WORKDEPT | PROJNO | PROJNAME |
|---|---|---|---|---|---|
| 000010 | HAAS | PRES | A00 | AD3100 | ADMIN SERVICES |
| 000010 | HAAS | PRES | A00 | MA2100 | WELD LINE AUTOMATION |
| 000020 | THOMPSON | MANAGER | B01 | PL2100 | WELD LINE PLANNING |
| 000030 | KWAN | MANAGER | C01 | IF1000 | QUERY SERVICES |
| 000030 | KWAN | MANAGER | C01 | IF2000 | USER EDUCATION |
| 000060 | STERN | MANAGER | D11 | MA2110 | W L PROGRAMMING |
| 000150 | ADAMSON | DESIGNER | D11 | MA2112 | W L ROBOT DESIGN |
| 000160 | PIANKA | DESIGNER | D11 | MA2113 | W L PROD CONT PROGS |
| 000220 | LUTZ | DESIGNER | D11 | MA2111 | W L PROGRAM DESIGN |

In the second command editor, code the query that list employee number, last name, job, work department, project number, and project name for all employees who are not President nor Managers.

***Result list:***

| EMPNO | LASTNAME | JOB | WORKDEPT | PROJNO | PROJNAME |
|---|---|---|---|---|---|
| 000150 | ADAMSON | DESIGNER | D11 | MA2112 | W L ROBOT DESIGN |
| 000160 | PIANKA | DESIGNER | D11 | MA2113 | W L PROD CONT PROGS |
| 000220 | LUTZ | DESIGNER | D11 | MA2111 | W L PROGRAM DESIGN |
| 000230 | JEFFERSON | CLERK | D21 | AD3111 | PAYROLL PROGRAMMING |
| 000250 | SMITH | CLERK | D21 | AD3112 | PERSONNEL PROGRAMMING |
| 000270 | PEREZ | CLERK | D21 | AD3113 | ACCOUNT PROGRAMMING |
| 000320 | MEHTA | FIELDREP | E21 | OP2011 | SCP SYSTEMS SUPPORT |
| 000330 | LEE | FIELDREP | E21 | OP2012 | APPLICATIONS SUPPORT |
| 000340 | GOUNOT | FIELDREP | E21 | OP2013 | DB/DC SUPPORT |

***Problem 2: (continued)***

In the third command editor, code the query that combines your first query from the command editor 1 and your second query from the command editor 2 using INTERSECT keyword.

**Explain the cursor (result table)**

**ANSWER:**

shows the employees who both work in departments A00 through D11 and are not Presidents or Managers. These employees appear in the result because they meet both conditions from the first and second queries. In this case, the result includes only Designers from department D11 who are assigned to specific projects. The INTERSECT keyword filters the data to include only those employees who are common in both result sets.

***Problem 3:***

Consider your INTERSECT query in problem #1. Replace the keyword INTERSECT with EXCEPT keyword.

**Explain the cursor (result table)**

**ANSWER:**

The result table shows the employee who is an ANALYST but does not have an education level of 18. In this case, only QUINTANA is an ANALYST with a different education level (16), so the result only shows QUINTANA. The EXCEPT keyword removes the employees who are in both lists and keeps only the ones from the first list that are not in the second.

***Problem 4:***

Consider your INTERSECT query in problem #2. Replace the keyword INTERSECT with EXCEPT keyword.

**Explain the cursor (result table)**

**ANSWER:**

This is because the EXCEPT keyword removes any employee that also appears in the second query, which excludes Presidents and Managers. So, the result only includes employees from the first query who are not found in the second query, meaning they are Presidents or Managers working in those departments.

## Exercise 10. SQL DQL – SQL VIEWS AND MERGE STATEMENTS

### What is this Exercise is About

This exercise provides a knowledge to work with VIEWS and MERGE statements.

### What You Should Be Able To Do

At the end of the lab exercises, you should be able to:

- Create views from existing tables in the Database.
- Code MERGE statement that combines an INSERT statement with an UPDATE or DELETE statement.

### Introduction

See the data model at the start of this exercise guide to get the table names, column names and descriptions for each table.

### Supporting Resources

Instructor's instructional materials

SQL References (Online resources, handouts, notes, etc.)

***Problem 1:***

Create a view called MYEMPVW1 that contains the data; employee number, last name, first name, work department, and salary for employees from employee table.

***Result list:***

. . . 42 row(s) in memory

***Problem 2:***

Create a view called MYEMPVW2 that contains the data; employee number, last name, first name, work department, and income for all employees working in departments A00, C01, and E11.
The income is the sum of the salary, bonus, and commission of the employee.

Name the view derived column TOTAl_INCOME.

***Result list:***

***Problem 3:***

Create a view called MYEMPVW3 that contains the data; employee number, last name, and the number of years the employee has served the company up to the current date. Create new column names for the view as EMP_ID, LAST_NAME, and YRS_SERVED respectively.

***Problem 4:***

Create a view called MYEMPVW4 that contains the data; employee number, last name, first name, and education level for employees from employee table. The view should contain only rows with education level greater than 17.

***Problem 5:***

Create a view called MYEMPVW5 that contains the data; employee number, last name, first name, and education level for employees from employee table. The view should restrict users to insert into the view an education level that is  less than 12.

***Problem 6:***

Create a view called MYEMPVW6 that contains the data from view MYEMPVW5. MYEMPVW6 should contain rows with an education level between 15 and 18.

**Exercise 11. XML AND XQUERY**

**What is this Exercise is About**

This exercise provides a knowledge to work with XML and XQUERY.

## What You Should Be Able To Do

At the end of the lab exercises, you should be able to:

Create XML tables.

Insert XML data into XML tables.

Code XQUERY statements to retrieve XML data from XML tables. Code XQUERY with FLWOR expression.

## Introduction

See the data model at the start of this exercise guide to get the table names, column names and descriptions for each table.

## Supporting Resources

Instructor's instructional materials

SQL References (Online resources, handouts, notes, etc.)

***Problem 1:***

Create an XML-enabled database named **`IM_XMLDB`**.

***Problem 2:***

Create an XML-enabled tables named **`xml_employee`** and **`xml_clients`**inside **`IM_XMLDB`**Database.

**XML_EMPLOYEE** table

| Column Name | Meaning | Data Type | NULLS Allowed |
|---|---|---|---|

| | | | |
|---|---|---|---|
| EMPNO | Employee number | SMALLINT | N |
| (**PK**) NAME | Name of employee | VARCHAR(30) | N |
| ADDRESS | Address of employee | VARCHAR(30) | N |
| JOB | Job Position of employee | **XML** | N |
| CONTACTINFO | Detailed contact information | **XML** | N |
| CIDNO (**FK**) | ID number of the client | SMALLINT | Y |

**XML_CLIENTS** table

| **Column Name** | **Meaning** | **Data Type** | **NULLS Allowed** |
|---|---|---|---|
| CIDNO (**PK**) | ID number of the client | SMALLINT | N |
| NAME | Name of the client | VARCHAR(30) | N |
| ADDRESS | Address of the client | **XML** | N |
| STATUS | Marital status | VARCHAR(10) | Y |
| CONTACTINFO | Detailed contact information | **XML** | N |

***Problem 3:***

Insert 5 rows/records into XML_EMPLOYEE table by using the data of your 5 classmates.

Insert 3 rows/records into XML_CLIENTS table by using the data of your 3 neighbours.

***Problem 4:***

List employee number, employee name, and job for all employees who served more than 5 years.

***Problem 5:***

List employee number, employee name, job, client name, and client contact information for all employees who served 1 or more clients.

***Problem 6:***

List employee job in hierarchical format for all employees.

***Problem 7:***

List client address in hierarchical format for all single clients.

***Problem 8:***

List client address and contact information in tabular format.

***Problem 9:***

List department name, department head, and position for all employees creating a XML element nodes for each column names respectively. List only those employees who served less than 5 years.

***Problem 10:***

Code XQuery that returns a hierarchical output for the employee's job.

***Problem 11:***

Code XQuery that returns a hierarchical output for the clients contact information.

***Problem 12:***

Code XQuery with FLWOR expression that returns a hierarchical output for the clients contact information. Set a WHERE clause to retrieved any client/s that matches your predicate.

***Problem 13:***

Code XQuery with FLWOR expression that returns HTML format of the column, job for all employees.

***Problem 14:***

Demonstrate XQuery with three-part conditional expression and XQuery with embedded SQL statement.